Using Ontologies to Evaluate Knowledge-based Systems

Michael Gruninger

NIST

Many systems have hidden assumptions about their domain that must be rendered explicit if we are to identify the principles on which they are based. Verified ontologies provide one way of making these assumptions explicit. A verified ontology consists of a specification of a class of mathematical structures together with a proof of two fundamental properties:

- **Satisfiability**: every structure in the class is a model of the ontology's axioms;
- **Axiomatizability**: every model of the ontology's axioms is isomorphic to some structure in the class.

Strictly speaking, we only need to show that a model exists in order to demonstrate that an ontology is satisfiable. However, in the axiomatization of ontologies, we need a complete characterization of the possible models. For example, if we are considering the domain of activities, occurrences, and timepoints, to show that a theory is satisfiable, we need only specify an occurrence of an activity that together with the axioms are satisfied by some structure. The problem with this approach is that we run the risk of having demonstrated satisfiability only for some restricted class of activities. For example, a theory of activities that supports scheduling may be shown to be consistent by constructing a satisfying interpretation, but the interpretation may require that resources cannot be shared by multiple activities or it may require all activities to be deterministic. Although such a model may be adequate for such activities, it would in no way be general enough for our purposes; we would want a comprehensive theory of activities that explicitly characterize the classes of activities, timepoints, objects, and other assumptions that are guaranteed to be satisfied by the specified structures.

When implementing knowledge-based systems, we are faced with the additional challenge that almost no existing software application has an explicitly axiomatized ontology. However, we can model a software application *as if* it were an inference system with an axiomatized ontology, and use this ontology to predict the set of sentences that the inference system decides to be satisfiable. This is the *Ontological Stance*, and is analogous to Dennett's intentional stance, which is the strategy of interpreting the behavior of an entity by treating it as if it were a rational agent who performs activities in accordance with some set of intentional constraints.

Using the ontological stance, we can define capabilities of knowledge-based systems and explain why certain techniques fail when extended to new domains. In particular, we can characterize the knowledge used in a given domain, and how this knowledge influences a particular reasoning task. Ontologies support a semantic assumption-based approach to tractable reasoning -- rather than identify syntactic classes of theories, we can reason about the assumptions that the ontology entails.

An example of this is the CardWorld Ontology that axiomatizes shape-based object recognition in scenes consisting of 2D surfaces with occlusion. The ontology allows characterization of the complexity of finding a model of an image together with the axioms of the ontology. In particular, tractable subclasses can be defined by

- Assumptions on images (e.g. accidental alignments)
- Assumptions on scenes (e.g. layered surfaces)
- Assumptions on depiction (e.g. errors in edge detection)